Claims Listing

1. (Previously Presented) A system that facilitates efficient code construction, comprising:

a processor for executing the following components:

a component that receives a first code designed in a noise model, the first code comprises algorithms utilized to correct noise errors with high probability, the first code is intended to refer to encoded data as well as error detection codes and includes a linear code, wherein the first code is generated based at least in part on a sequence of messages; and

a transformation component that transforms the first code to a new code that has essentially same length parameters as the first code but is hidden to a computationally bounded adversary, the transformation component utilizes a random number generator to perform algebraic transformations on data utilizing the first code to generate the new code, and the transformation component hides the first code *via* randomizing data that employs the first code thereby not enabling the computationally bounded adversary to determine a location of critical bits to attack, wherein the new code acts as a protective wrapping of the first code, such that an attack on the new code by the computationally bounded adversary would appear as a noise attack on the first code, as the attack would be randomly distributed across the first code and not concentrated on a particular location within the first code, this allows the first code to act as it was designed to and utilize the algorithms to correct the noise errors with a high success rate;

a decoder that determines the first code from the new code, the decoder

accesses algorithms utilized by the transformation component to decode the new code and determine the first code, wherein the decoder knows the sequence of messages *via* the decoder being synchronized with the transformation component;

a tracing component that determines whether a user accessing the first code is a valid user *via* a unique watermark associated with a particular user and embedded in the first code, wherein if the watermark does not correlate to an authorized user, access is denied; and

a pseudo random number generator, the pseudo random number generator generates two pseudo random numbers a and b, each n number of bits, based upon a position within a sequence of one of the messages, and further generates a random permutation σ that permutes the *n* bits.

2. (Original) The system of claim 1, the new code appears random to the computationally bounded adversary.

3. (Canceled)

4. (Original) The system of claim 1, the transformation component comprises a pseudo-random number generator that facilitates transforming the first code into the new code.

5. (Canceled)

6. (Previously Presented) The system of claim 1, the decoder comprising a checking component that determines whether the first code has been corrupted.

7. (Original) The system of claim 6, the checking component utilizing a checking function *h*: $\Sigma^n \rightarrow \{0,1\}$, where Σ is a finite alphabet that defines a family of codes and *n* is a length parameter for Σ.

8. (Original) The system of claim 6, the checking component outputting a vector, the first code being corrupted when the vector is a non-zero vector.

9. (Previously Presented) The system of claim 1, the decoder utilizes a unique decoding function *g*, where $g(\tilde{c}) = c$ when $d(c, \tilde{c}) < \frac{d}{2}$,and *c* is a code word, $\tilde{c}$ is code word *c* that has been altered, and *d* is a Hamming distance between any two code words.

10. (Previously Presented) The system of claim 1, the decoder utilizes a list decoding function *g*, where g($\tilde{c}$) = L ,where $\tilde{c}$ is a codeword *c* that has been altered, and L is a list of code words that contain *c.*

11 – 13. (Canceled)

14. (Previously Presented) The system of claim 1, the transformation component sends a randomized code word to the decoder, the randomized code word

having the form $a \times \sigma(f(m_i)) + b$, where *f* is an encoding function, *m* is a message, *i* is the position of the message within the sequence, and *x* is a bitwise multiplication operator.

15. (Previously Presented) The system of claim 14, the transformation component embeds information relating to the sequence of messages into the new code.

16. (Original) The system of claim 15, the first code has a length of n_l, and the information relating to the sequence of messages embedded in n_l locations in the new code.

17. (Canceled)

18. (Previously Presented) The system of claim 16, an encoder sending the new code to the decoder, the new code having embedded therein the seed.

19. (Canceled)

20. (Previously Presented) A system that hides a codeword from a computationally bounded adversary, comprising:

a processor for executing the following components:

a code generator that generates a first code designed in a noise model and based at least in part upon a sequence of messages that are desirably relayed to a receiver, the first code comprising algorithms utilized to correct noise errors with high probability;

a code hiding module that creates a second code, the second code being a pseudo random version of the first code, the second code appears to be random to a computationally bounded adversary; and

a decoder that determines the first code from the second code, wherein the decoder knows the sequence of messages *via* the decoder being synchronized with the code generator,

wherein the second code acts as a protective wrapping of the first code, such that an attack on the second code by the computationally bounded adversary would appear as a noise attack on the first code, this allows the first code to utilize the algorithms to correct the noise errors;

a tracing component that determines whether a user accessing the first code is a valid user *via* a unique watermark associated with a particular user and embedded in the first code; and

a pseudo random number generator, the pseudo random number generator generates two pseudo random numbers *a* and b, each *n* number of bits, based upon a position within a sequence of one of the messages, and further generates a random

permutation σ that permutes the *n* bits.

21. (Original) The system of claim 20, further comprising an encoding component that encodes a message and creates a code word, the encoding component encodes the message with a code that has a minimum relative distance ε and rate $1 - K\varepsilon$ for some constant ***K*** > 1.

22. (Original) The system of claim 21, further comprising a component that utilizes the encoded message and divides the encoded message into a number of blocks *B*, the *B* blocks being of substantially similar size.

23. (Original) The system of claim 22, the plurality of blocks encoded using (*n, k, n -k + 1*) Reed-Solomon code, where n is a resulting size of the encoded blocks and *k* is a size of the blocks prior to encoding.

24. (Original) The system of claim 23, the code hiding module comprising a bipartite expander graph with a number of edges being substantially similar to *Bn*, and symbols within the *B* blocks are randomly assigned an edge within the bipartite expander graph.

25. (Original) The system of claim 20, the decoder comprises one or more algorithms that facilitate solving a minimum vertex cover problem.

26. (Original) The system of claim 20, further comprising a synchronization

component that synchronizes the code generator with the decoder.

27. (Original) The system of claim 20, the code hiding module embeds synchronization information into the second code.

28. (Currently Amended) A method for hiding a data package from a computationally bounded adversary, comprising:

receiving a message that is desirably transferred to an authorized user;

encoding the message utilizing an encoding scheme designed in a noise model;

algebraically transforming the encoded message into a first code, the first code rendered random to [[an]]<u>the</u> unauthorized user and based at least in part upon a sequence of messages that are desirably relayed to a receiver, and the first code comprising algorithms utilized to correct noise errors;

transforming the first code to a second code that has essentially same length parameters as the first code but is hidden to a computationally bounded adversary, wherein the second code acts as a protective wrapping of the first code, such that an attack on the second code by the computationally bounded adversary would appear as a noise attack on the first code;

utilizing the algorithms of the first code to correct the noise errors;

determining whether a user accessing the first code is a valid user via a unique watermark associated with a particular user and embedded in the first code; and

generating two pseudo random numbers *a* and *b*, each *n* number of bits, based upon a position within a sequence of one of the messages, and further

generating a random permutation σ that permutes the n bits.

29. (Original) The method of claim 28, further comprising decoding the message, wherein the message is decoded at least in part by solving a minimum vertex cover problem.

30. (Original) The method of claim 28, further comprising embedding information into the first code relating to the message's position within a sequence of messages.

31. (Original) The method of claim 28, further comprising decoding the first code based at least in part upon knowledge of the message's position within a sequence of messages.

32. (Canceled)

33. (Currently Amended) A system that facilitates efficient code construction, the system comprising:

a memory; and

a processor ~~for executing the following~~coupled to the memory, the processor configured to perform acts comprising:

~~means for~~ receiving a first code designed in a noise model and based at least in part upon a sequence of messages that are desirably relayed to a receiver, the first code comprises algorithms utilized to correct noise errors;

~~means for~~ transforming the first code into a second code, the second code

appearing random to a computationally bounded adversary and having substantially similar length as the first code, the means for transforming utilizes a random number generator to perform algebraic transformations on data utilizing the first code to generate the second code;

~~means for~~ decoding the second code to obtain the first code, wherein the means for decoding the second code is based at least in part upon knowledge of the message's position within a sequence of messages[[;]], wherein the second code acts as a protective wrapping of the first code, such that an attack on the second code by the computationally bounded adversary would appear as a noise attack on the first code;

~~means for~~ utilizing the algorithms of the first code to correct the noise errors;

~~means for~~ determining whether a user accessing the first code is a valid user *via* a unique watermark associated with a particular user and embedded in the first code; and

~~means for~~ generating two pseudo random numbers *a* and b, each *n* number of bits, based upon a position within a sequence of one of the messages, and further means for generating a random permutation σ that permutes the *n* bits.

34. (Currently Amended) A computer storage media having computer executable instructions stored thereon [[to]] which when executed by a processor perform acts comprising:

transform a first code into a second code, the second code being a pseudo-randomized version of the first code and having essentially a same length as the first code, the second code appearing truly random to a computationally bounded adversary, [[and]]wherein the first code is designed in a noise model and comprises algorithms utilized to correct noise errors;

~~wherein synchronization information is embedded~~ embed synchronization information into the second code, ~~wherein the first code is designed in a noise model and comprises algorithms utilized to correct noise errors,~~ and wherein the second code acts as a protective wrapping of the first code, such that an attack on the second code by the computationally bounded adversary would appear as a noise attack on the first code;

~~wherein the first code designed in the noise model~~ utilize[[s]] the algorithms to correct the noise errors; [[and]]

~~wherein the first code comprises a tracing component that~~ determine[[s]] whether a user accessing the first code is a valid user ***via*** a unique watermark associated with a particular user; [[and]]

~~a pseudo random number generator, the pseudo random number generator~~ generate[[s]] two pseudo random numbers a and b, each n number of bits, based upon a position within a sequence of one of the messages[[,]]; and

~~further~~ generate[[s]] a random permutation σ that permutes the n bits.

35. (Currently Amended) A computer storage media having a data structure stored thereon [[that]] which when executed by a processor performs acts comprising:

receive[[s]] a first code that is designed in a noise model [[and]];

transform[[s]] the first code into a second code, the second code being a substantially similar size as the first code and appearing random to a computationally bounded adversary, wherein the second code acts as a protective wrapping of the first code, such that an attack on the second code by the computationally bounded adversary would appear as a noise attack on the first code[[, and]];

~~wherein synchronization information is embedded~~ embed synchronization information into the second code;

~~wherein the first code designed in the noise model~~ utilize[[s]] algorithms to correct the noise errors; [[and]]

~~wherein a tracing component is embedded~~ embed a tracing component in the first code [[that]], the tracing component determines whether a user accessing the first code is a valid user via a unique watermark associated with a particular user; [[and]]

~~a pseudo random number generator, the pseudo random number generator~~ generate[[s]] two pseudo random numbers a and b, each n number of bits, based upon a position within a sequence of one of the messages[[,]]; and

~~further~~ generate[[s]] a random permutation σ that permutes the *n* bits.